

## SENDGRID INTEGRATION WITH PYTHON

Date	04 Nov 2022
Team ID	PNT2022TMID12569
Project Name	CUSTOMER CARE REGISTRY

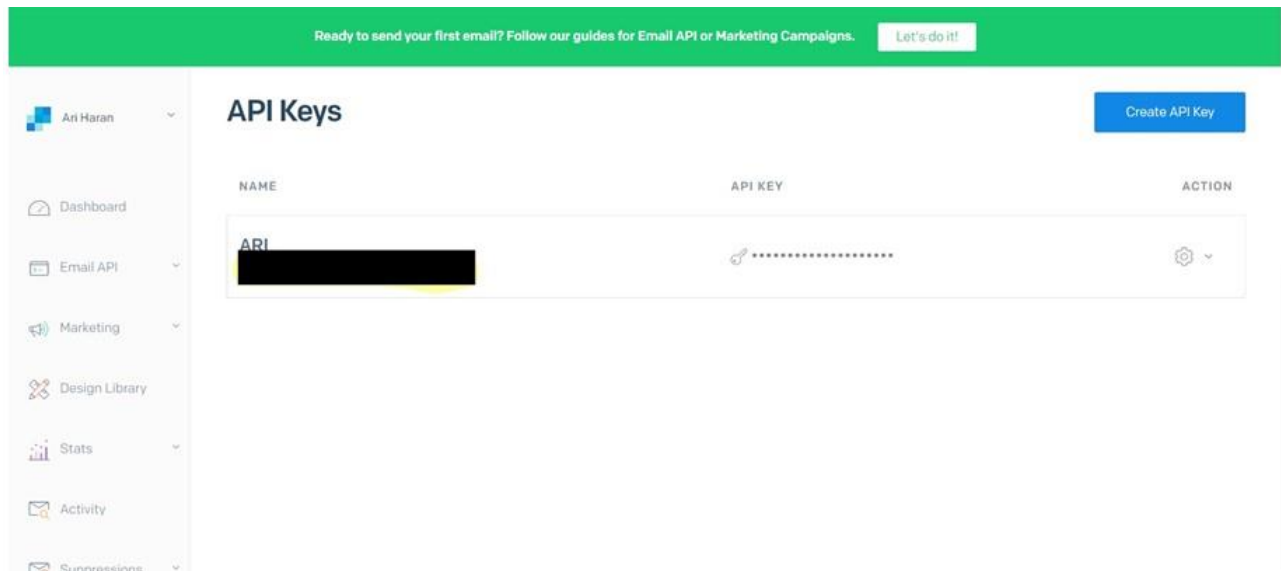
### STEP 1:

#### REQUIREMENTS:

**Python 2.6, 2.7, 3.4 or 3.5.**

### STEP 2:

Create an API key



### STEP 3:

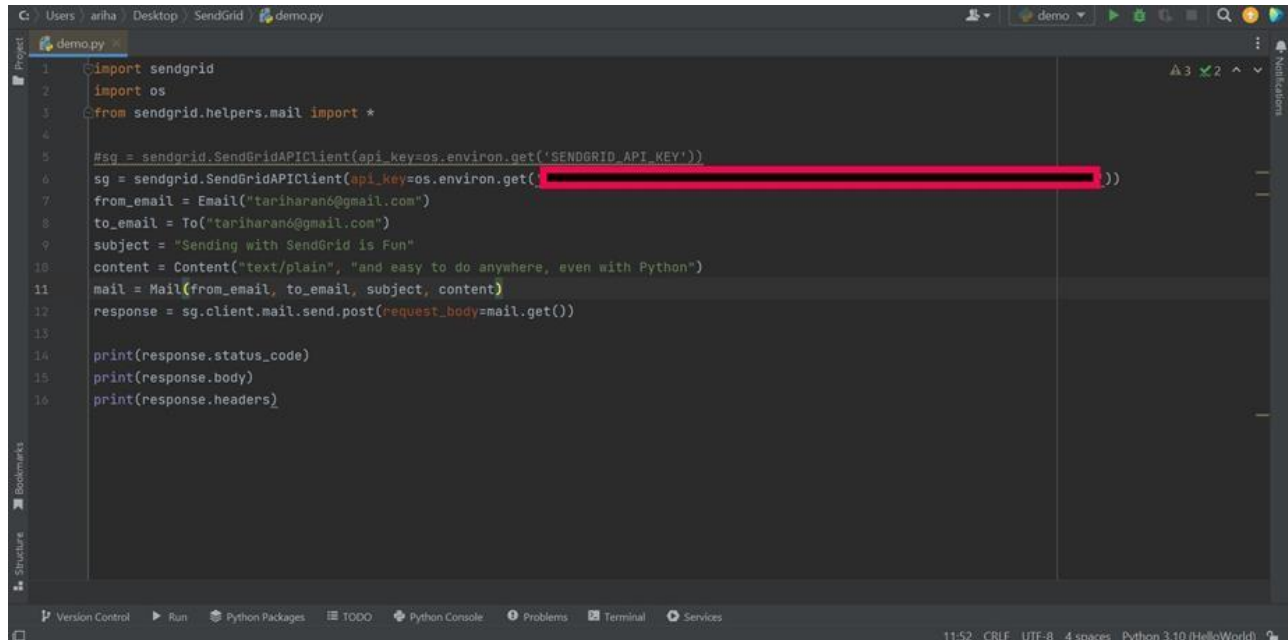
INSTALL

PACKAGE:

> pip install sendgrid

SETP 4:

## SEND EMAIL



```
1 import sendgrid
2 import os
3 from sendgrid.helpers.mail import *
4
5 #sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
6 sg = sendgrid.SendGridAPIClient(api_key=os.environ.get('SENDGRID_API_KEY'))
7 from_email = Email("tariharan@gmail.com")
8 to_email = To("tariharan@gmail.com")
9 subject = "Sending with SendGrid is Fun"
10 content = Content("text/plain", "and easy to do anywhere, even with Python")
11 mail = Mail(from_email, to_email, subject, content)
12 response = sg.client.mail.send.post(request_body=mail.get())
13
14 print(response.status_code)
15 print(response.body)
16 print(response.headers)
```

**SENDGRID PYTHON CODE :**

```

1  import os
2  from sendgrid import SendGridAPIClient
3  from sendgrid.helpers.mail import Mail
4
5  message = Mail(
6      from_email='from_email@example.com',
7      to_emails='to@example.com',
8      subject='Sending with Twilio SendGrid is Fun',
9      html_content='<strong>and easy to do anywhere, even with
Python</strong>') 10
try:
11     sg = SendGridAPIClient(os.environ.get('SENDGRID_API_KEY'))
12     response = sg.send(message)
13     print(response.status_code)
14     print(response.body) 15     print(response.headers) 16 except Exception as
e:
17     print(e.message)

```

## HTTP CLIENT PROGRAM:

```

1  """HTTP Client library"""
2  import json
3  import logging
4  from .exceptions import handle_error
5
6  try:
7      # Python 3
8      import urllib.request as urllib
9      from urllib.parse import urlencode
10     from urllib.error import HTTPError 11
except ImportError:
12     # Python 2

```

```
import urllib2 as urllib
```

```
from urllib2 import HTTPError
```

```

15     from urllib import urlencode
16
17 _logger = logging.getLogger( name )
18
19
20 class Response(object):
21     """Holds the response from an API call.""" 22
22
23     def init (self, response):
24         """
25         :param response: The return value from a
26         open call
27         on a urllib.build_opener()
28         :type response: urllib response object
29         """
30         self._status_code = response.getcode()
31         self._body = response.read()
32         self._headers = response.info()
33
34     @property
35     def status_code(self):
36         """
37         :return: integer, status code of API call
38         """
39         return self._status_code
40
41     @property
42     def body(self):
43         """
44         :return: response from the API
45         """
46         return self._body
47
48     @property

```



```

49         """
50         :return: dict of response headers
51         """
52         return self._headers
53
54     @property
55     def to_dict(self):
56         """
57         :return: dict of response from the API
58         """
59         if self.body:
60             return json.loads(self.body.decode('utf-8'))
61         else:
62             return None
63
64
65 class Client(object):
66     """Quickly and easily access any REST or REST-like API.""" 67
68     # These are the supported HTTP verbs
69     methods = {'delete', 'get', 'patch', 'post', 'put'} 70
71     def init (self,
72 host,
73 request_headers=None,
74 version=None,
75 url_path=None,
76 append_slash=False, 77 timeout=None):
78         """
79         :param host: Base URL for the api. (e.g.
80             https://api.sendgrid.com)
81         :type host: string
82         :param request_headers: A dictionary of the headers you want

```

applied on all calls

:type request\_headers: dictionary

request\_headers



```

84         :param version: The version number of the
                        API.

85         Subclass _build_versioned_url for custom
                        behavior.

86         Or just pass the version as part of the URL
87         (e.g. client._("/v3"))

88         :type version: integer

89         :param url_path: A list of the url path
                        segments

90         :type url_path: list of strings

91         """

92         self.host = host

93         self.request_headers = request_headers or
                        {}

94         self._version = version

95         # _url_path keeps track of the dynamically
                        built url

96         self._url_path = url_path or []

97         # APPEND SLASH set

98         self.append_slash = append_slash

99         self.timeout = timeout

100

101     def _build_versioned_url(self, url):
102         """Subclass this function for your own needs.
103         Or just pass the version as part of the URL
104         (e.g. client._('/v3'))

105         :param url: URI portion of the full URL being requested

106         :type url: string

107         :return: string

108         """

109         return '{}{}/v{}{}'.format(self.host, str(self._version),
                                    url)

110

111     def _build_url(self, query_params):

```

---

```
112
```

---

```
    """Build the final URL to be passed to urllib
```

113

114           :param query\_params: A dictionary of all the query



```

115         :type query_params: dictionary
116         :return: string
117         """
118         url = ''
119         count = 0
120         while count < len(self._url_path):
121             url += '{}/{}'.format(self._url_path[count])
122             count += 1
123
124         # add slash
125         if self.append_slash:
126             url += '/'
127
128         if query_params:
129             url_values = urlencode(sorted(query_params.items()),
130 True)
131
132             url = '{}?{}'.format(url, url_values)
133
134         if self._version:
135             url = self._build_versioned_url(url)
136         else:
137             url = '{}{}'.format(self.host, url)
138             return url
139
140     def _update_headers(self, request_headers):
141         """Update the headers for the request
142
143         :param request_headers: headers to set for the API call
144         :type request_headers: dictionary
145         :return: dictionary

```



```
145         self.request_headers.update(request_headers)
146
147     def _build_client(self, name=None):
```

"""Make a new Client object





```

150         :param name: Name of the url segment
151         :type name: string
152         :return: A Client object
153         """
154         url_path = self._url_path + [name] if name
155         else self._url_path
156         return Client(host=self.host,
157                       version=self._version,
158                       request_headers=self.request_headers,
159                       url_path=url_path,
160                       append_slash=self.append_slash,
161                       timeout=self.timeout)
162
163     def _make_request(self, opener, request, timeout=None):
164         """Make the API call and return the response. This is
165         separated into
166         it's own function, so we can mock it easily for testing.
167
168         :param opener:
169         :type opener:
170         :param request: url payload to request
171         :type request: urllib.Request object
172         :param timeout: timeout value or None
173         :type timeout: float
174         :return: urllib response
175         """
176         timeout = timeout or self.timeout
177         try:
178             return opener.open(request, timeout=timeout)
179         except HTTPError
180         as err:
181             exc = handle_error(err)
182             exc.cause = None
183             _logger.debug('{method} Response: {status}

```



```
{body}'.format(
```



```

182         status=exc.status_code,
183         body=exc.body))
184         raise exc
185
186     def _(self, name):
187         """Add variable values to the url.
188         (e.g. /your/api/{variable_value}/call)
189         Another example: if you have a Python reserved word,
190         such as global,
191         in your url, you must use this method.
192
193         :param name: Name of the url segment
194         :type name: string
195         :return: Client object
196         """
197         return self._build_client(name)
198
199     def getattr (self, name):
200         """Dynamically add method calls to the url, then
201         call a method.
202
203         (e.g. client.name.name.method())
204         You can also add a version number by using
205         .version(<int>)
206
207         :param name: Name of the url segment or method
208         call
209
210         :type name: string or integer if name ==
211         version
212
213         :return: mixed
214         """
215         if name == 'version':
216             def get_version(*args, **kwargs):

```



```
210             :param args: dict of settings
211             :param kwargs: unused
```

```
return: string version
```



|||||





```

214         self._version = args[0]
215         return self._build_client()
216         return get_version
217
218         # We have reached the end of the method chain, make the
219         # API call
220         if name in self.methods:
221             method = name.upper()
222
223             def http_request(
224                 request_body=None,
225                 query_params=None,
226                 request_headers=None,
227                 timeout=None,
228                 **_):
229                 """Make the API call
230
231                 :param timeout: HTTP request timeout. Will be
232                 propagated to
233                 urllib client
234                 :type timeout: float
235                 :param request_headers: HTTP headers. Will be
236                 merged into
237                 current client object state
238                 :type request_headers: dict
239                 :param query_params: HTTP query parameters
240                 :type query_params: dict
241                 :param request_body: HTTP request body
242                 :type request_body: string or json-serializable
243                 object
244                 :param kwargs:
245                 :return: Response object
246                 """
247                 if request_headers:

```



```

243         self._update_headers(request_headers)
244
245     if request_body is None:
246         data = None
247     else:
248         # Don't serialize to a JSON formatted
249         # str
250         # if we don't have a JSON Content-Type
251         if 'Content-Type' in
252         self.request_headers and \
253         self.request_headers['Content-Type'] !=
254         \
255         'application/json':
256         data = request_body.encode('utf-8')
257         else:
258         self.request_headers.setdefault(
259         'Content-Type', 'application/json')
260         data =
261         json.dumps(request_body).encode('utf-8')
262
263     opener = urllib.build_opener()
264     request = urllib.Request(
265     self._build_url(query_params),
266     headers=self.request_headers,
267     data=data,
268     )
269     request.get_method = lambda: method
270
271     _logger.debug('{method} Request: {url}'.format(
272     method=method,
273     url=request.get_full_url()))
274     if request.data:
275         _logger.debug('PAYLOAD: {data}'.format(
276         data=request.data))
277         _logger.debug('HEADERS: {headers}'.format(

```









```
276         response = Response(
277             self._make_request(opener, request, timeout=timeout)
278         )
279
280         _logger.debug('{method} Response: {status}
281                        {body}'.format(
282                            method=method,
283                            status=response.status_code,
284                            body=response.body))
285
286         return response
287
288     return http_request 288
289
290     else:
291         # Add a segment to the URL
292         return self._(name)
293
294     def getstate (self):
295         return self. dict_
296
297     def setstate (self, state):
```



